Description

METHOD AND SYSTEM FOR ACCELERATION OF SECURE SOCKET LAYER TRANSACTIONS IN A NETWORK

Field of the Invention

This invention is concerned with accelerating secure transactions within a network.

Background of the Invention

The Secure Sockets Layer (SSL) protocol was developed by Netscape™ to enable the secure transmission of data over TCP/IP networks. SSL (now also known as Transport Layer Security (TLS) since the Internet Engineering Task Force (IETF) has taken over responsibility for the SSL standard) is commonly used to support secure transactions on the World Wide Web (Web). As more and more financial and confidential transactions are conducted using the Web, the ability to secure these transactions using SSL is increasingly important.

SSL supports multiple applications. The protocol runs above TCP/IP and below the application layer, which includes protocols such as the HyperText Transport Protocol (HTTP), the Internet Messaging Access Protocol (IMAP), the Simple Mail Transfer Protocol (SMTP), and the File Transfer Protocol (FTP). The SSL protocol consists of a set of routines for providing security services such as authentication and encryption.

Referring to Fig. 1, when a secure webpage is requested by a Web browser (block 10), such as (Netscape NAVIGATOR) or Microsoft Internet Explorer, the request is received at a server at TCP port 443 (unsecured session requests are received at TCP port 80). The server then sends the browser its digital certificate (block 12). The browser then checks the digital certificate (block 14). Provided the certificate is valid, the browser and server then negotiate a session key (block 16). The

secure channel is established and all data transmitted over that channel is encrypted with the session key (block 18). When the browser receives the encrypted webpage, it decrypts it using the session key (block 20).

There is a high processing cost associated with providing security via SSL transactions. Authentication and encryption in secure transactions both require much more processing power than is required in non-secure transactions. This processing requirement can affect the performance of servers responding to requests for secure transactions; this effect is noticeable to Web users due to the increased amount of time that may be required to conduct secure transactions. Hardware accelerators which off-load the tasks of establishing an SSL session and encrypting/decrypting data from a server to the accelerator are widely available, though they are not employed at all servers which handle requests for secure webpages.

Even if hardware SSL accelerators are used to reduce the amount of time required to complete a secure transaction, the requests and responses sent from the client and server are still likely to be affected by factors that create network bottlenecks and slow the delivery of Webpages in the network. These factors include: slow servers, modem and network latency, and the bandwidth of the communication pipe.

It would advantageous to provide a transparent software solution to SSL acceleration that could be employed at the client. It would also be advantageous to provide a solution to SSL acceleration which could be combined with other approaches to reducing the bandwidth necessary to deliver SSL webpages as well as reducing communication latency within the network.

Summary of the Invention

These needs have been met by a system and method of accelerating SSL webpages in which a client proxy associated with a client browser rewrites links to secure websites in a webpage requested by the client browser before the page is returned to the client browser; the links are rewritten from their original format such that they are recognized and processed as requests for SSL webpages by another proxy in the network, in one embodiment a device intermediating between the client and server. If a secure website is requested, the request is recognized by the other proxy which returns the request to its original format before requesting the page. The proxy establishes an SSL session with the server and decrypts and compresses the response before sending it to the client proxy, where the response is scanned and any links to secure webpages are rewritten before the response is returned to the client. This approach is transparent to the client.

In other embodiments, this approach to SSL acceleration may be combined with other solutions to reduce bandwidth and communication latency, for instance, by using certain compression techniques and network architectures.

Brief Description of the Drawings

Fig. 1 is a flowchart showing the prior art approach to establishing and conducting an SSL session.

Fig. 2 is a block diagram showing a potential network configuration in accordance with the invention.

Fig. 3 is a flowchart showing acceleration of SSL transactions in accordance with the invention.

Detailed Description

In Fig. 2, a client device 22 (such as a personal computer or other computing device) having a Web browser 24, such as Netscape NAVIGATOR or Microsoft Internet Explorer, and software acting as a client proxy 26, is connected via a network connection 28 to a device 30 intermediating between the client and a server 34 in the network 28. (In other embodiments, the client proxy may be running on another machine.) The device 30 may be a server or any other computing device. The device 30 is running specialized software 32, discussed in greater detail below, which enables the device 30 to handle requests for secure Webpages from the client 22 and then process the webpage received from the server 34 as required before returning the webpage to the client proxy 26; this software 32 may also decrypt and compress the webpage before returning it to the client proxy 26. In other embodiments, the device or server may be associated with hardware SSL accelerators. The server 34 contains content 36 which is requested by the client 22 (the content 36 may be stored at the server or at a storage device associated with the server 34).

In one embodiment, the client 22 and device 30 are members of a private network, while the server 34 is a member of a public network. In other embodiments, the client 22 is as member of both the private and public networks. In one embodiment, disclosed in U.S. patent application serial number 10/012,743, filed December 7, 2001, which is herein incorporated by reference, the client proxy 26 relays requests from the client 22 to the device 30, which then sends the request to the server 34. The device 30 may contain a cache of content retrieved from the server; the cached content, if current, may be used to assemble at least part of the reply to request for content.

In another embodiment, disclosed in U.S. patent application serial number 10/012,743, the private network is a persistently-connected caching network featuring multiple hubs, or network devices, which are capable of caching material transmitted through the hub as material is sent either from a server or another caching hub in response to a client's request for the material. The network devices may employ a socket layer capable of combining multiple messages from different machines, threads, and/or processes into single TCP/IP packets to be relayed along message hubs in the persistent network. Due to the direct connection between dedicated socket pairs of network members, there is bi-directional asynchronous communication between the network members.

The acceleration of SSL websites is achieved by having the intermediating device, rather than the client, retrieve the secure webpage from the server, and then decrypting and compressing the secure webpage, using either known or proprietary compression techniques, before sending the response to the client proxy.

In Fig. 3, the client proxy scans a received webpage (block 38) to determine whether the webpage contains any links to secure webpages (block 40). Secure webpages are indicated, for instance, by the presence of "https," indicating the use of secure http, in the URL. Any links to secure webpages are rewritten so that the intermediating device can recognize the request is for a secure webpage (block 60). The link can be rewritten from its original format to indicate a request for a secure webpage in several ways. In one embodiment, an https request can be rewritten as an http request as follows: https://www.bank.com/x is rewritten as http://propelsecure.www.bank.com/x. In another embodiment, the https request can be redirected to a subdomain indicating a request for a secure webpage as

follows: https://www.bank.com/ is rewritten as https://www.bank.com/propel. Once links to secure webpages in the webpage have been rewritten (block 60), or if there are no links that need to be rewritten (block 40), the webpage is returned to the client's browser (block 42).

A secure webpage is requested by the browser via the rewritten link in the webpage (block 44). This request is sent to the client proxy which sends it on to the intermediating device. The intermediating device receives the request for the webpage (block 46). Where the request from the client is an https request, the client proxy and the intermediating device have to form a secure connection. When the request from the client is an http request, no secure connection needs to be formed. When the client proxy and intermediating device are members of a private network, the private network provides a greater level of security than the public network, so data sent between the server and client proxy outside of an SSL connection is less likely to be compromised than it would be if it were sent over a public network.

Since the links to secure webpages are rewritten as subdomains or controlled domains, any cookies previously sent by a content server to the client will still be sent with the rewritten request. Cookies remain attached to all requests which are passed to the client proxy and the intermediating device.

The device returns the request to its original format (block 48) and requests the secure webpage from the server (block 50). The device and the server establish a secure connection (block 52) and the server sends the secure webpage to the intermediating device (block 54). The intermediating device decrypts the webpage and compresses it (block 56).

Any type of compression scheme may be used. In one embodiment, disclosed in U.S. patent application serial number 10/012,743, which was earlier incorporated by reference, text or pictures are compressed into one or more unique codes, or identifiers, typically 64-bit hash codes. When text is compressed, the text is broken up in one embodiment through use of an HTML parser which breaks on certain HTML tags; in other embodiments, text can be broken up by words or paragraphs. The identifiers and content associated with the identifiers are stored at a database at the encoder (here, the proxy). Where identifiers have been seen in sequence previously by the encoder, that sequence of identifiers is consolidated into a new identifier. The identifiers are then sent to the client proxy, which is associated with a database or cache containing identifiers and content previously received from the encoder (proxy). If an identifier is in the client proxy's database, the client proxy is able to decompress the identifier; otherwise, the client proxy requests the content associated with the identifier from the encoder (proxy). This request-reply sequence is recursive and continues until the decoder at the client proxy is able to decompress the requested data.

In one embodiment, a page template may be created and cached at both the intermediary device and the client proxy. In this instance, provided the page template has not been updated, only dynamic material differs each time a page is requested; if the page template has changed, it will be updated. This could be particularly useful, for instance, if a client frequently requests financial information, such as a bank balance or information about stocks, that is likely to change over relatively short periods of time. While the specific data is likely to change, the underlying page displaying the data probably does not change very much over time.

Therefore, if the static elements of the page are compressed and cached, only the dynamic information needs to be sent to the client proxy.

In other embodiments, disclosed in U.S. patent application serial number 10/012,743, the encoder will send uncompressed content along with an identifier when there is no record at the encoder of the identifier being sent to the client proxy. In still other embodiments, other known compression schemes, such as LZW compression, may be used.

Referring again to Fig. 3, the intermediary device sends the compressed webpage to the client proxy (block 58) where it is decompressed. The client proxy scans the webpage for any links to secure webpages (block 38) and rewrites these links before returning the webpage to the client's browser.